Springboard DSC Program

Capstone 2 Milestone report

Screening for Dyslexia Using Eye Tracking During Reading

Using Eye Movement during Reading to predict Dyslexia

By Harinee Madhusudhan

June 2020

Eye tracking has been gaining in popularity over the past decade as a window into observers' visual and cognitive processes. This project uses eye movement data from dyslexic kids to predict the possibility of dyslexia based on measured eye movements. The source for this data is available at https://doi.org/10.6084/m9.figshare.c.3521379.v1

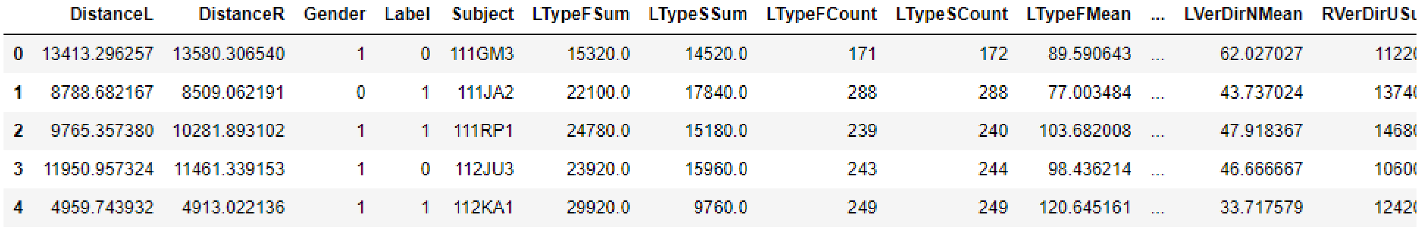
Our study is based on a sample of 97 high-risk subjects with early identified word decoding difficulties and a control group of 88 low-risk subjects. These subjects were selected from a larger population of 2165 school children attending second /third grade.

The objective was to create simple and commonly used machine learning models like logistic regression, and maybe others to compare how they perform regarding for the task of predicting the high risk of long- term reading disabilities.

Description of the dataset, how obtained, cleaned, and wrangled it

By tracking eye movements during reading, the observant was able to follow the reading process as it occurs in real-time and obtain objective measurements of this process. The data being sampled provide a next to continuous record of reading that reflects both the speed and accuracy of the processes involved.

Read the processed file with the features - the file contains one row per subject, 185 subjects and 52 features per subject. The data quality is high. In the previous modules the data has been tested for the data to not contain any missing values in any of the columns.



	DistanceL	DistanceR	Gender	Label	Subject	LTypeFSum	LTypeSSum	LTypeFCount	LTypeSCount	LTypeFMean	...	LVerDirNMean	RVerDirUSu
0	13413.296257	13580.306540	1	0	111GM3	15320.0	14520.0	171	172	89.590643	...	62.027027	1122
1	8788.682167	8509.062191	0	1	111JA2	22100.0	17840.0	288	288	77.003484	...	43.737024	1374
2	9765.357380	10281.893102	1	1	111RP1	24780.0	15180.0	239	240	103.682008	...	47.918367	1468
3	11950.957324	11461.339153	1	0	112JU3	23920.0	15960.0	243	244	98.436214	...	46.666667	1060
4	4959.743932	4913.022136	1	1	112KA1	29920.0	9760.0	249	249	120.645161	...	33.717579	1242

Dependent Variable

The column label is the prediction column. This is the dependent variable. 0 is control group and 1 is dyslexic group.

Independent Variables

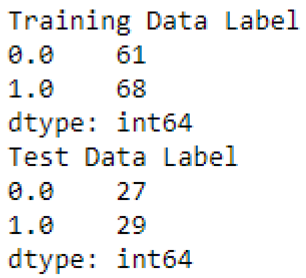
there are 52 independent variables. All but gender are numerical/decimal variables. The gender column is a categorical variable and has 1 for males and 0 for females

Eye Movement Analysis

The dataset has 88 subjects in the control group and 97 subjects in the dyslexic group. This way the data seems to be very balanced. use StandardScaler to scale the variables to a standard normal format.

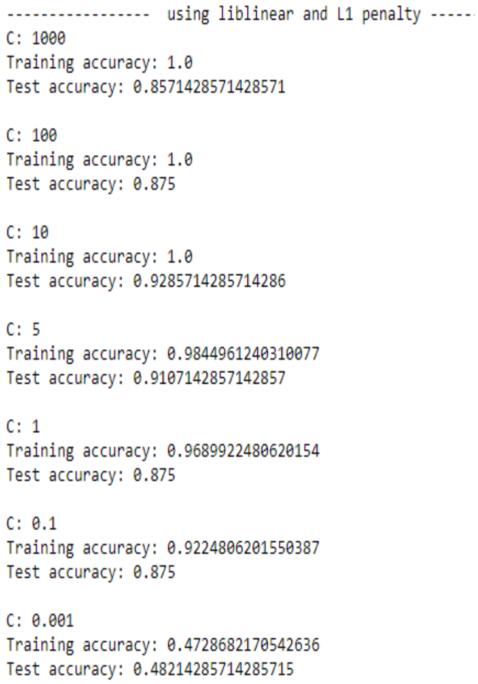
A training and validation approach is used. the available data is split with 70% of the data used for training the model and 30% of the data used to validate the model as the test data set. The dependent and independent variables are split into X_train and y_train for the training data set and X_test and y_test for the testing dataset. A stratified ratio preserved rows are split across the test and training data. The original data has a control/dyslexia ratio of (88/97) 90%,

the training data has a ratio of (61/68) 90% and the test data has a ratio of (27/29) 93%.

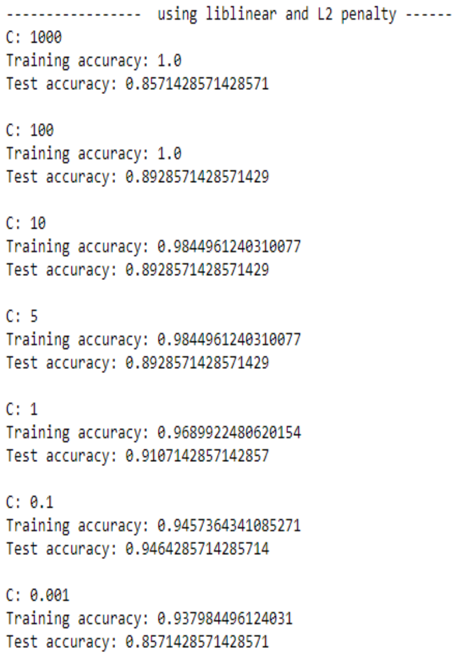


```
Training Data Label
0.0    61
1.0    68
dtype: int64
Test Data Label
0.0    27
1.0    29
dtype: int64
```

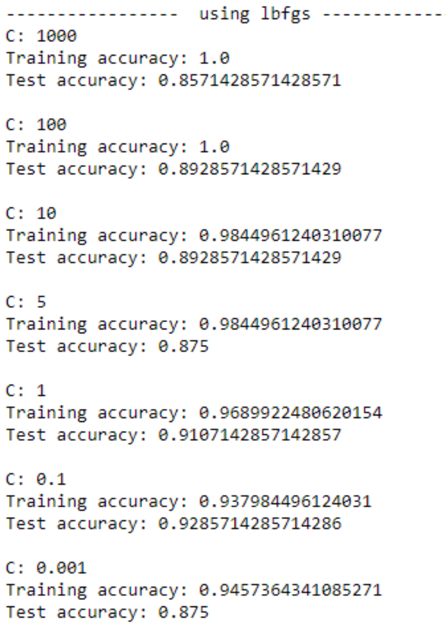
Now to find the hyperparameters for a logistic regression model, we try with different C values, 1000 through 0.001. Looking at the results a C value of 10 provides the best training and test accuracy.



```
----------------  using liblinear and L1 penalty -----
C: 1000
Training accuracy: 1.0
Test accuracy: 0.8571428571428571

C: 100
Training accuracy: 1.0
Test accuracy: 0.875

C: 10
Training accuracy: 1.0
Test accuracy: 0.9285714285714286

C: 5
Training accuracy: 0.9844961240310077
Test accuracy: 0.9107142857142857

C: 1
Training accuracy: 0.9689922480620154
Test accuracy: 0.875

C: 0.1
Training accuracy: 0.9224806201550387
Test accuracy: 0.875

C: 0.001
Training accuracy: 0.4728682170542636
Test accuracy: 0.48214285714285715
```

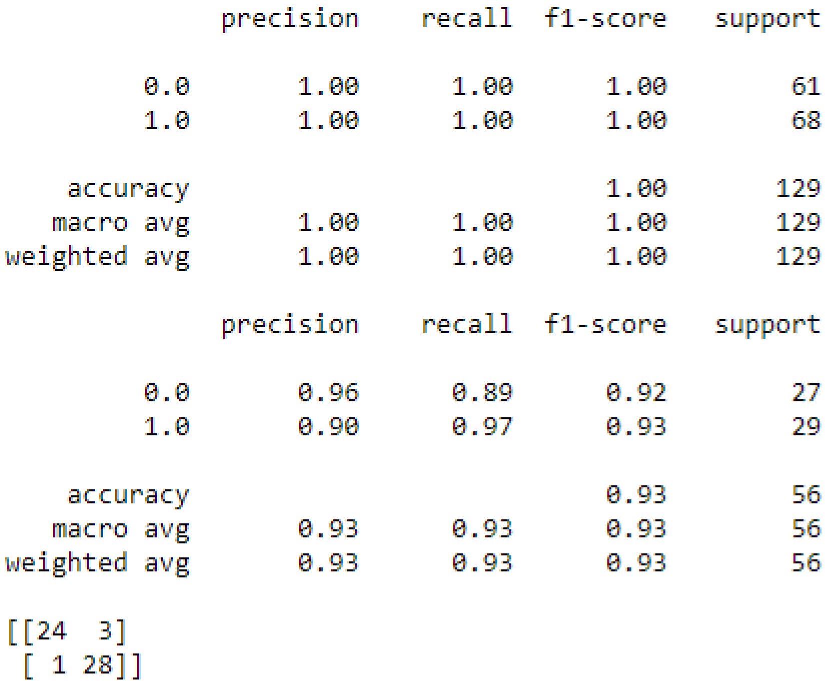


```
----------------  using liblinear and L2 penalty ------
C: 1000
Training accuracy: 1.0
Test accuracy: 0.8571428571428571

C: 100
Training accuracy: 1.0
Test accuracy: 0.8928571428571429

C: 10
Training accuracy: 0.9844961240310077
Test accuracy: 0.8928571428571429

C: 5
Training accuracy: 0.9844961240310077
Test accuracy: 0.8928571428571429

C: 1
Training accuracy: 0.9689922480620154
Test accuracy: 0.9107142857142857

C: 0.1
Training accuracy: 0.9457364341085271
Test accuracy: 0.9464285714285714

C: 0.001
Training accuracy: 0.937984496124031
Test accuracy: 0.8571428571428571
```



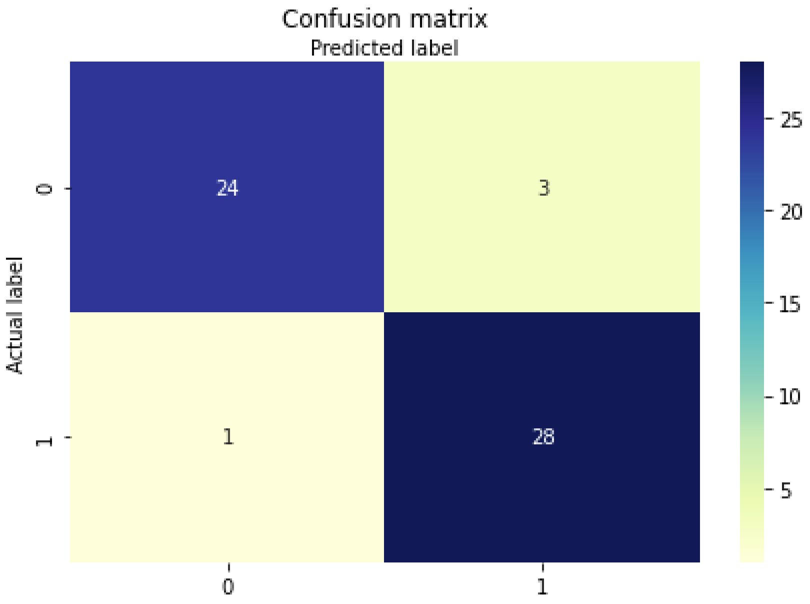
```
-----------------  using lbfgs ------------
C: 1000
Training accuracy: 1.0
Test accuracy: 0.8571428571428571

C: 100
Training accuracy: 1.0
Test accuracy: 0.8928571428571429

C: 10
Training accuracy: 0.9844961240310077
Test accuracy: 0.8928571428571429

C: 5
Training accuracy: 0.9844961240310077
Test accuracy: 0.875

C: 1
Training accuracy: 0.9689922480620154
Test accuracy: 0.9107142857142857

C: 0.1
Training accuracy: 0.937984496124031
Test accuracy: 0.9285714285714286

C: 0.001
Training accuracy: 0.9457364341085271
Test accuracy: 0.875
```

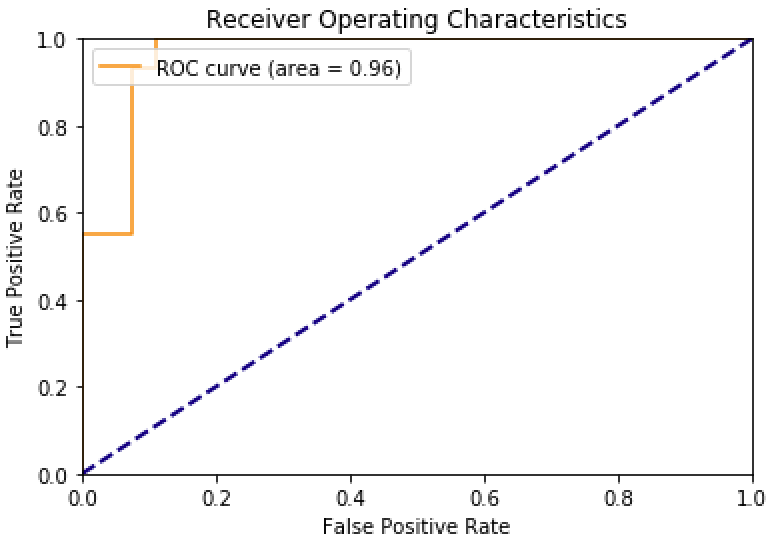
We try to fit a Logistic regression model with a regularization parameter (C) of 10 and a maximum of 500 iterations.



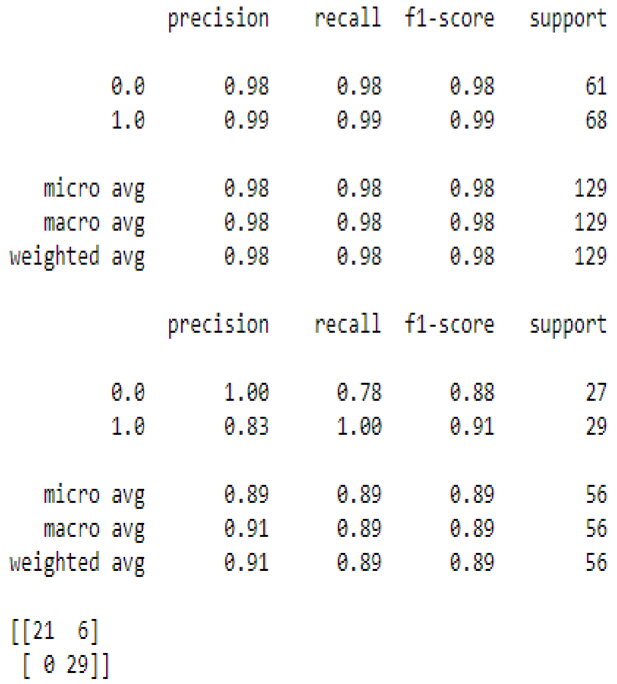
```
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        61
         1.0       1.00      1.00      1.00        68

    accuracy                           1.00       129
   macro avg       1.00      1.00      1.00       129
weighted avg       1.00      1.00      1.00       129

              precision    recall  f1-score   support

         0.0       0.96      0.89      0.92        27
         1.0       0.90      0.97      0.93        29

    accuracy                           0.93        56
   macro avg       0.93      0.93      0.93        56
weighted avg       0.93      0.93      0.93        56

[[24  3]
 [ 1 28]]
```

Then test the model using predictions from the test data set. We prepare and print the classification report for the training and test data.

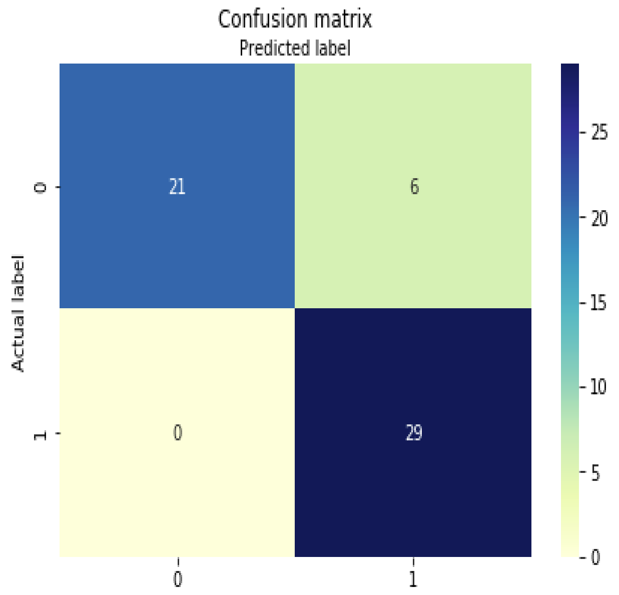


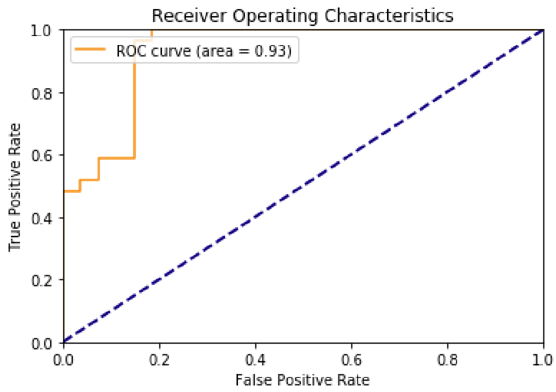


For the model = Logistic Regression(solver='lbfgs') the results are:

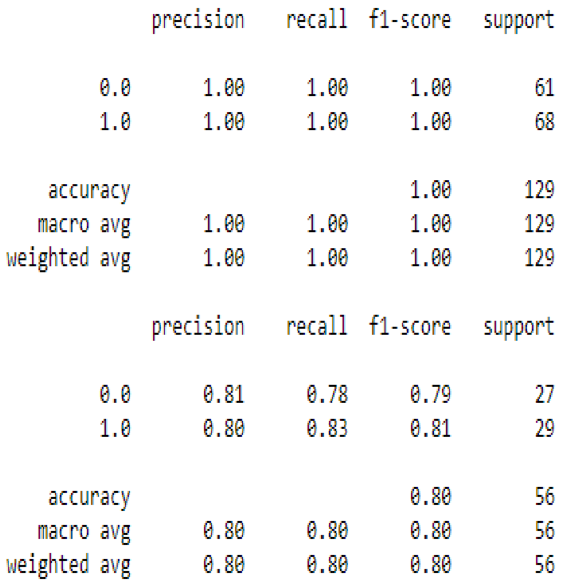


```
              precision    recall  f1-score   support

         0.0       0.98      0.98      0.98        61
         1.0       0.99      0.99      0.99        68

   micro avg       0.98      0.98      0.98       129
   macro avg       0.98      0.98      0.98       129
weighted avg       0.98      0.98      0.98       129

              precision    recall  f1-score   support

         0.0       1.00      0.78      0.88        27
         1.0       0.83      1.00      0.91        29

   micro avg       0.89      0.89      0.89        56
   macro avg       0.91      0.89      0.89        56
weighted avg       0.91      0.89      0.89        56

[[21  6]
 [ 0 29]]
```

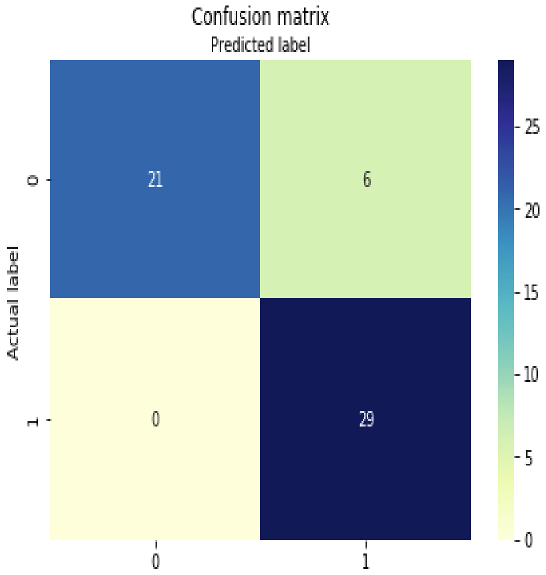


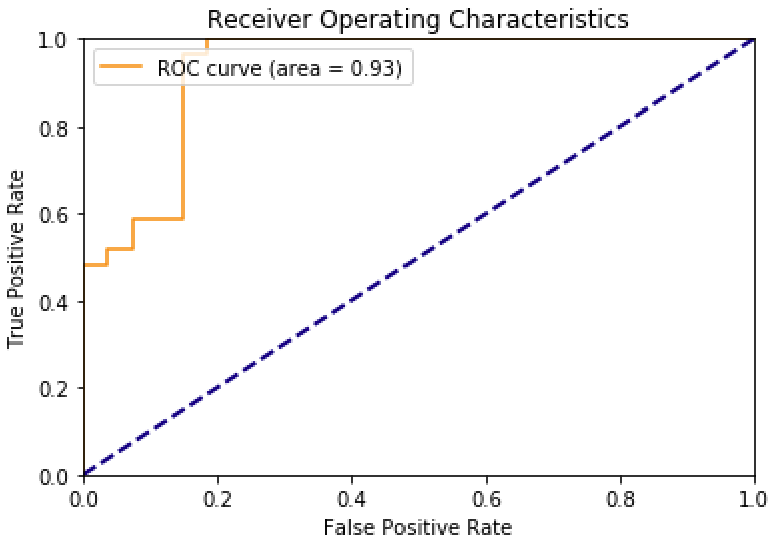


Now trying with the Gradient Boosting Classifier, to see if there are any better results.



```
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        61
         1.0       1.00      1.00      1.00        68

    accuracy                           1.00       129
   macro avg       1.00      1.00      1.00       129
weighted avg       1.00      1.00      1.00       129
              precision    recall  f1-score   support

         0.0       0.81      0.78      0.79        27
         1.0       0.80      0.83      0.81        29

    accuracy                           0.80        56
   macro avg       0.80      0.80      0.80        56
weighted avg       0.80      0.80      0.80        56
```





It looks like the linear regression with solver = liblinear and L1 penalty provides the best results

Future proposal:

Even though it has been a known fact that the eye movements of dyslexic readers are different from those of typical readers, usually research has focused almost exclusively on identifying group-level differences. The objective here is to use machine learning and predictive modeling, to individual-level predictions with high sensitivity and specificity.

The algorithm analyzes can be expanded to the tracking signal sample by sample and switches between two other mutually exclusive states: distortions, transients other than fixations, and saccades which was used for computation here. To conclude using eye tracking during reading may prove very useful in early identification of individuals in need of support.